Time-Critical Distributed Visualization with Fault Tolerance

Jinzhu Gao  University of Minnesota, Morris
Huadong Liu  University of Tennessee, Knoxville
Jian Huang  University of Tennessee, Knoxville
Micah Beck  University of Tennessee, Knoxville
Qishi Wu  University of Memphis
Terry Moore  University of Tennessee, Knoxville
James Kohl  Oak Ridge National Laboratory
Current Challenges

- Massive data must be visualized with high efficiency…

Richtmyer-Meshkov Turbulent Simulation
274 time steps, each is $2048 \times 2048 \times 1920$

3D Core-Collapse Supernova Simulation
300 time steps, each is $864 \times 864 \times 864$
(Courtesy image of Ross Toedte, SciDAC TSI project)
Our Goal

- A fault-tolerant time-critical visualization system that tolerates
  - Heterogeneity of processors
  - Perils of wide-area distribution across the Internet
Our Goal
Our Method

- LoD data selection based on a general importance metric
- Dynamic scheduling scheme with fault-tolerance
Importance Metric

- Assign different time budget for different regions based on their importance

- The importance of a block is based on its contribution to the final image

\[ I = w_{\text{app}} \times I_{\text{app}} + w_{\text{val}} \times I_{\text{val}} + w_{\text{view}} \times I_{\text{view}} \]

- $I_{\text{app}}$: Application-dependent factor
- $I_{\text{val}}$: Value-dependent factor
- $I_{\text{view}}$: View-dependent factor
- $w$: Weight coefficient
The importance of a block may depend on the underlying applications.

For example:

- Time-critical applications: choose the highest possible resolution for a region

\[ I_{app} = Height_{root} - Height_{node} \]
Importance – Value-dependent

\[ I_{val} = w_{opa} \times V_{opa} + w_{var} \times V_{var} + w_{serr} \times (1 - V_{serr}) \]

- \( V_{opa} \): Opaqueness of a block
- \( V_{val} \): Value variance of a block
- \( V_{serr} \): Spatial error of a block
- \( w \): Weight coefficient
Importance – View-dependent

- The importance of a block may depend on the eye position

\[ I_{\text{view}} = 1 - \frac{ID_{\text{traversal}}}{N_{\text{block}}} \]

- An invisible block doesn’t have an importance value

- \( ID_{\text{traversal}} \): sequential order during front-to-back traversal
- \( N_{\text{block}} \): total number of blocks
Dynamic Fault-Tolerance Load Balancing

**Master-Worker model:**

- **Worker processors:**
  - Distributed and heterogeneous depots
    - "Depot": a processing unit with local storage and computing resources
  - Perform rendering tasks

- **Master processor:**
  - The client’s local machine
  - Schedules entire parallel run and composites the final image
Dynamic Fault-Tolerance Load Balancing

Major tasks:

- Adaptive scheduling of rendering tasks
- Dynamic scheduling of data movement
- Dealing with faults
- Quality-driven back-off
Adaptive Scheduling of Rendering Tasks

- Two generic data structures:
  - A dynamically ranked pool of depots
    - The depots are ranked in the order of their estimated rendering time for a task
  - A two-level priority queue of tasks
    - High priority queue (HPQ):
      - Tasks ready to be assigned
      - Primary key: importance value
      - Secondary key: optimal task processing time
    - Low priority queue (LPQ):
      - Tasks assigned to one or more depots but not finished
        - Key: estimated time left for completion
  - In HPQ and LPQ, tasks are sorted using their keys in a decreasing order
Adaptive Scheduling of Rendering Tasks

- $D_1$ becomes available $\rightarrow$ $T_j$ is assigned to it
- $D_2$ becomes available $\rightarrow$ It tries to help out with tasks in LPQ
- $D_3$ becomes available $\rightarrow$ $T_i$ is assigned to it
Dealing with Faults

- Promote the failed task in LPQ back to HPQ
- A majority voting scheme to avoid incorrect computation result
Quality-driven Back-off

- To meet the user-specified time limit, several tasks that operate on high resolution data would be replaced with one task that operates on lower resolution data.

Tasks marked with a ‘*’ will not be rendered.
Test Environment

- 160 depots from the PlanetLab project and 10 depots from the National Logistical Networking Testbed (NLNT)

- A 128 time-step subset of the TSI data
  - Spatial resolution: 864×864×864
  - After data partition, multiresolution data generation, and 3-way replication: ~1TB of data was stored
Performance Evaluation

- Data preparation: **10-20 hours**
- Software raycasting is used
- About **51** seconds to process four time steps and generate an **800×800** image for each time step
  - It took **62 minutes** to perform the same task on a dedicated node with **2.2GHz P4 CPU, 512 KB cache** and **2GB RAM**
Performance Evaluation

The number of original blocks and visible blocks after culling at resolution level 0, 1, and 2 of a TSI dataset.
Logarithmic plot of the number of blocks rendered at different resolution level with different running deadline
Performance Evaluation

- Time-critical & fault tolerance
  - Initially 8 depots were used, deadline = 31 seconds
    - 1181 level-0 blocks can be rendered
  - If one depot is disabled,
    - 1025 level-0 blocks
    - 156 level-0 blocks are replaced by 32 level-1 blocks
  - If two depots are disabled
    - 876 level-0 blocks
    - 305 level-0 blocks are replaced by 52 level-1 blocks
Conclusion

- Perform time-critical visualization on hundreds of geographically distributed, free, unreserved, heterogeneous processors

- Demonstrate a great potential to use distributed heterogeneous processors as a fundamental computing platform
Acknowledgement

- NSF ACI-0329323, NGS-0437508
- Dr. Mark Duchaineau (Lawrence Livermore National Lab)
- Anthony Mezzacappa (ORNL), John Blondin (NCSU), Ross Toedte (ORNL) and the DOE SciDAC TeraScale Supernova Initiative project team
If you have any questions ...

- Please contact
  - Jinzhu Gao (gaoj@morris.umn.edu)
  - Huadong Liu (hliu@cs.utk.edu)
  - Jian Huang (huangj@cs.utk.edu)
Thank you!